BSTZ No. 42P16807
Express Mail No. EV323392952US

UNITED STATES PATENT APPLICATION

FOR

A PLATFORM AND METHOD FOR ESTABLISHING TRUST WITHOUT REVEALING IDENTITY

Inventor:

Ernie F. Brickell

Prepared by:

Blakely, Sokoloff, Taylor & Zafman LLP
12400 Wilshire Boulevard, Suite 700
Los Angeles, California 90025
(714) 557-3800

A PLATFORM AND METHOD FOR ESTABLISHING TRUST WITHOUT REVEALING IDENTITY

Field

Embodiments of the invention generally relate to secured communications, namely a platform and method for establishing that information came from a trusted hardware device without revealing information concerning the identity of the trusted hardware device.

General Background

For many modern communication systems, the reliability and security of exchanged information is a significant concern. To address this concern, the Trusted Computing Platform Alliance(TCPA) developed security solutions for platforms. In accordance with a TCPA specification entitled "Main Specification Version 1.1b," published on or around February 22, 2002, each personal computer (PC) is implemented with a trusted hardware device referred to as a Trusted Platform Module (TPM). Each TPM contains a unique endorsement key pair (EK), which features a public EK key (PUBEK) and a private EK key (PRIVEK). The TPM typically has a certificate for the PUBEK signed by the manufacturer.

During operation, the TPM records information about the software and hardware environment of its PC. In order for an outside party (referred to as a "challenger") to learn about the software and/or hardware environment of the PC, a challenger can request the TPM to generate and provide a report. This creates two opposing security concerns.

First, the challenger needs to be sure that the report is really coming from a valid TPM. Second, the owner of the PC

wants to maintain as much privacy as possible. In particular, the owner of the PC wants to be able to give reports to different challengers without those challengers being able to determine that the reports are coming from the same TPM.

One proposed solution to these security issues is to establish a Trusted Third Party (TTP). For instance, the TPM would create an Attestation Identify Key pair (AIK), namely a public AIK key and a private AIK key. The public AIK key would be placed in a certificate request signed with the PRIVEK, and subsequently sent to the TTP. The certificate for the PUBEK would also be sent to the TTP. The TTP would check that the signed certificate request is valid, and if valid, the TTP would issue a certificate to the TPM. The TPM would then use the public AIK and the TTP issued certificate when the TPM received a request from a challenger. Since the AIK and certificate would be unrelated to the EK, the challenger would get no information about the identity of the TPM or PC implemented with the TPM.

In practice, the above-identified approach is problematic because it requires TTPs to be established. Identifying and establishing various parties that can serve as TTPs has proven to be a substantial obstacle.

Another proposed solution is set forth in a co-pending U.S. Application No. 10/306,336, which is also owned by the assignee of the present application. This technique utilizes two interactive proofs (IP1, IP2). Thus, in order to achieve a probability of cheating to be less than 1 in 2^{20}, the TPM would need to complete twenty (20) modular exponentiations with a 2048 bit modulus and a 2000-bit exponent for IP1, and twenty (20) modular exponentiations with a 2048-bit modulus and a 160-bit exponent for IP2. Since a TPM may require forty-five (45)

seconds to compute a single modular exponentiation with a 2048-bit modulus and a 2000-bit exponent, the efficiency of the TPM computations has proven to be a substantial obstacle as well.

BRIEF DESCRIPTION OF THE DRAWINGS

Features and advantages of embodiments of the invention will become apparent from the following detailed description of the invention in which:

Figure 1 illustrates a system featuring a platform implemented with a Trusted Platform Module (TPM) that operates in accordance with one embodiment of the invention.

Figure 2 illustrates a first embodiment of the platform including the TPM of Figure 1.

Figure 3 illustrates a second embodiment of the platform including the TPM of Figure 1.

Figure 4 illustrates an exemplary embodiment of a computer implemented with the TPM of Figure 2.

Figure 5 illustrates a flow diagram of a procedure to setup a TPM during manufacturing according to one embodiment of the invention.

Figure 6 illustrates a flow diagram of a procedure to setup each platform manufactured according to one embodiment of the invention.

Figure 7 illustrates a first exemplary embodiment of an interactive direct proof method for a platform (responder) to prove to a challenger that it knows authentication information without revealing that information.

Figure 8 illustrates a second exemplary embodiment of an interactive direct proof method for a platform (responder).

Figure 9 illustrates a third exemplary embodiment of an interactive direct proof method for a platform (responder).

Figure 10 illustrates a first exemplary embodiment of a non-interactive direct proof method for a platform (responder) to prove to a challenger that it knows authentication information without revealing that information.

DETAILED DESCRIPTION

Embodiments of the invention set forth in the following detailed description generally relate to secured communications. Herein, at least one embodiment of the invention relates to a system, device and method for proving that received information came from a trusted hardware device without revealing information concerning the identity of the trusted hardware device or stored cryptographic information. This is accomplished without the use of a Trusted Third Party (TTP). Rather, it is accomplished by a "direct proof" methodology in which computations by the TPM involve exponentiations using exponents of fixed length and small bit length (e.g., 160 bits). The bit length of each exponent is substantially less than one-half the bit length of a modulus associated with the exponentiation (e.g., one-third or less, normally one-eighth or less).

For one embodiment of the invention, the functionality of the TPM, which is configured to prove to a challenger that information (e.g., cryptographic key, digital signature, digital certificate, etc.) came from the TPM, is deployed as firmware. However, it is contemplated that such functionality may be deployed as dedicated hardware or software. Instructions or code forming the firmware or software are stored on a machine-readable medium.

Herein, "machine-readable medium" may include, but is not limited to a floppy diskette, hard disk, optical disk (e.g., CD-ROMs, DVDs, mini-DVDs, etc.), magneto-optical disk, semiconductor memory such as read-only memory (ROM), random access memory (RAM), any type of programmable read-only memory (e.g., programmable read-only memory "PROM", erasable

programmable read-only memories "EPROM", electrically erasable programmable read-only memories "EEPROM", or flash), magnetic or optical cards, or the like. It is contemplated that a signal itself and/or a communication link can be regarded as machine-readable medium since software may be temporarily stored as part of a downloaded signal or during propagation over the communication link.

In the following description, certain terminology is used to describe certain features of one or more embodiments of the invention. For instance, "platform" is defined as any type of communication device that is adapted to transmit and receive information. Examples of various platforms include, but are not limited or restricted to computers, personal digital assistants, cellular telephones, set-top boxes, facsimile machines, printers, modems, routers, or the like. A "communication link" is broadly defined as one or more information-carrying mediums adapted to a platform. Examples of various types of communication links include, but are not limited or restricted to electrical wire(s), optical fiber(s), cable(s), bus trace(s), or wireless signaling technology.

A "challenger" refers to any entity (e.g., person, platform, system, software, and/or device) that requests some verification of authenticity or authority from another entity. Normally, this is performed prior to disclosing or providing the requested information. A "responder" refers to any entity that has been requested to provide some proof of its authority, validity, and/or identity. A "device manufacturer," which may be used interchangeably with "certifying manufacturer," refers to any entity that manufactures or configures a platform or device (e.g., a Trusted Platform Module).

As used herein, to "prove" or "convince" a challenger that a responder has possession or knowledge of some cryptographic information (e.g., digital signature, a secret such as a key, etc.) means that, based on the information and proof disclosed to the challenger, there is a high probability that the responder has the cryptographic information. To prove this to a challenger without "revealing" or "disclosing" the cryptographic information to the challenger means that, based on the information disclosed to the challenger, it would be computationally infeasible for the challenger to determine the cryptographic information.

Such proofs are hereinafter referred to as direct proofs. The term "direct proof" refers to zero-knowledge proofs, as these types of proofs are commonly known in the field.

Throughout the description and illustration of the various embodiments of the invention discussed hereinafter, coefficients, variables, and other symbols (e.g., "h") are referred to by the same label or name. Therefore, where a symbol appears in different parts of an equation as well as different equations or functional description, the same symbol is being referenced.

I. GENERAL ARCHITECTURE

Referring now to Figure 1, an exemplary embodiment of a system featuring a platform implemented with a trusted hardware device (referred to as "Trusted Platform Module" or "TPM") that operates in accordance with one embodiment of the invention is shown. A first platform 102 (Challenger) transmits a request 106 that a second platform 104 (Responder) provides information about itself. In response to request 106, second platform 104 provides the requested information 108.

Additionally, for heightened security, first platform 102 may need to verify that requested information 108 came from a device manufactured by either a selected device manufacturer or a selected group of device manufacturers (hereinafter referred to as "device manufacturer(s) 110"). For instance, for one embodiment of the invention, first platform 102 challenges second platform 104 to show that it has cryptographic information (e.g., a signature) generated by device manufacturer(s) 110. The challenge may be either incorporated into request 106 (as shown) or a separate transmission. Second platform 104 replies to the challenge by providing information, in the form of a reply, to convince first platform 102 that second platform 104 has cryptographic information generated by device manufacturer(s) 110, without revealing the cryptographic information. The reply may be either part of the requested information 108 (as shown) or a separate transmission.

In one embodiment of the invention, second platform 104 comprises a Trusted Platform Module (TPM) 115. TPM 115 is a cryptographic device that is manufactured by device manufacturer(s) 110 and conforms to the operations of the protocols described in Figures 7-10. In one embodiment of the invention, TPM 115 comprises a processor with a small amount of on-chip memory encapsulated within a package. TPM 115 is configured to provide information to first platform 102 that would enable it to determine that a reply is transmitted from a valid TPM. The information used is content that would not make it likely that the TPM's or second platform's identify can be determined.

Figure 2 illustrates a first embodiment of second platform 104 with TPM 115. For this embodiment of the invention, second platform 104 comprises a processor 202 coupled to TPM 115. In

general, processor 202 is a device that processes information. For instance, in one embodiment of the invention, processor 202 may be implemented as a microprocessor, digital signal processor, micro-controller or even a state machine. Alternatively, in another embodiment of the invention, processor 202 may be implemented as programmable or hard-coded logic, such as Field Programmable Gate Arrays (FPGAs), transistor-transistor logic (TTL) logic, or even an Application Specific Integrated Circuit (ASIC).

Herein, second platform 104 further comprises a storage unit 206 to permit storage of cryptographic information such as one or more of the following: keys, hash values , signatures, certificates, etc. As shown below, a hash value of "X" may be represented as "Hash(X)". Of course, it is contemplated that such information may be stored within internal memory 220 of TPM 115 in lieu of storage unit 206 as shown in Figure 3. The cryptographic information may be encrypted, especially if stored outside TPM 115.

Figure 4 illustrates a specific embodiment of second platform 104 including a computer 300 implemented with TPM 115 of Figure 2. Computer 300 comprises a bus 302 and a processor 310 coupled to bus 302. Computer 300 further comprises a main memory unit 304 and a static memory unit 306.

Herein, main memory unit 304 is volatile semiconductor memory for storing information and instructions executed by processor 310. Main memory 304 also may be used for storing temporary variables or other intermediate information during execution of instructions by processor 310. Static memory unit 306 is non-volatile semiconductor memory for storing information and instructions for processor 310 on a more permanent nature. Examples of static memory 306 include, but are not limited or

restricted to read only memory (ROM). Both main memory unit 304 and static memory unit 306 are coupled to bus 302.

In one embodiment of the invention, computer 300 further comprises a data storage device 308 such as a magnetic disk or optical disc and its corresponding drive may also be coupled to computer 300 for storing information and instructions.

Computer 300 can also be coupled via bus 302 to a display 314, such as a cathode ray tube (CRT), Liquid Crystal Display (LCD) or any flat panel display, for displaying information to an end user. Typically, an alphanumeric input device 316 (e.g., keyboard, keypad, etc.) may be coupled to bus 302 for communicating information and/or command selections to processor 310. Another type of user input device is cursor control unit 318, such as a mouse, a trackball, touch pad, stylus, or cursor direction keys for communicating direction information and command selections to processor 310 and for controlling cursor movement on display 314.

A communication interface unit 320 is also coupled to bus 302. Examples of interface unit 320 include a modem, a network interface card, or other well-known interfaces used for coupling to a communication link forming part of a local or wide area network. In this manner, computer 300 may be coupled to a number of clients and/or servers via a conventional network infrastructure, such as a company's Intranet and/or the Internet, for example.

It is appreciated that a lesser or more equipped computer than described above may be desirable for certain implementations. Therefore, the configuration of computer 300 will vary from implementation to implementation depending upon numerous factors, such as price constraints, performance

requirements, technological improvements, and/or other circumstances.

II. PLATFORM SET-UP

Figure 5 illustrates the setup performed for each platform class according to one embodiment of the invention. A "platform class" may be defined by the device manufacturer to include one or more types of platforms or devices. For instance, a platform class may be the set of all platforms that have the same security relevant information. This security relevant information could contain some of the information that is included in the EK or AIK certificate in the TCPA model. It could also include the manufacturer and model number of the particular platform or device.

For each platform class, a device manufacturer creates the cryptographic parameters that the manufacturer uses for that platform class. The device manufacturer creates a signature key that it uses to sign the secrets for the devices (e.g., platform 104 or TPM 115) that it manufactures.

In one embodiment of the invention, the device manufacturer utilizes a public key cryptographic function (e.g., RSA function) to create an RSA public/private key pair with public modulus n, public exponent e, and private exponent d (block 402). The public key is based on values e,n while the private key is based on d,n. This can be created using well known methods, such as those described in Applied Cryptography, by Bruce Schneier, John Wiley & Sons; ISBN: 0471117099; Second Edition (1996). The modulus n should be chosen large enough so that it is computationally infeasible to factor n.

The device manufacturer specifies a parameter Z, which is an integer between zero (0) and n (block 404).

The device manufacturer specifies a security parameter W, which is an integer between zero (0) and n (block 406). However, picking W too small or too large may introduce a security failure. In one embodiment of the invention, W is selected to be approximately 2^{160}. Selecting W to be between 2^{80} and the square root of n is recommended.

In one embodiment of the invention, the device manufacturer computes a prime number P, such that P = u*n+1 (block 408). Any value of u can be used; however, to retain an acceptable level of security, the value P should be large enough so that computing a discrete logarithm "mod P" is computationally infeasible.

The device manufacturer generates a Platform Class Certificate that comprises cryptographic parameters e, n, u, P, Z, W, the security relevant information of the platform class, and the name of the device manufacturer (block 410). In one embodiment, the parameters u and P would not both be included since given n and one of these parameters, the other can be computed by P = u*n + 1.

In one embodiment of the invention, the device manufacturer uses the same cryptographic parameters e, n, u, P, W for several different platform classes, and just varies the value Z for the different platforms. In this case, the values of Z may be chosen to differ by approximately or at least 4W, although the selected difference is a design choice.

Once the Platform Class Certificate is generated, the device manufacturer provides the Platform Class Certificate to

the platforms or devices it manufactures which belong to that particular platform class (block 412).

The distribution of cryptographic parameters associated with the Platform Class Certificate from a responder (e.g., second platform 104 in Fig. 1) to a challenger may be accomplished in a number of ways. However, these cryptographic parameters should be distributed to the challenger in such a way that the challenger is convinced that the Platform Class Certificate was generated by the device manufacturer.

For instance, one accepted method is by distributing the parameters directly to the challenger. Another accepted method is by distributing the Platform Class Certificate signed by a certifying authority, being the device manufacturer as one example. In this latter method, the public key of the certifying authority should be distributed to the challenger, and the signed Platform Class Certificate can be given to each platform in the platform class. The responder can then provide the signed Platform Class Certificate to the challenger.

Figure 6 illustrates the setup performed for a platform (responder) manufactured according to one embodiment of the invention. The TPM of the responder platform chooses a random number m such that $0 < m-Z < W$ (block 502). The TPM may blind this random number m before sending it to the certifying manufacturer for signature (block 504). This blinding operation is performed to obfuscate the exact contents of the random number m from the certifying manufacturer. In this case, the TPM chooses a random number, B, where $1 < B < n-1$ (block 506), and computes $A = B^e \bmod n$ (block 508). Then, the TPM computes $m' = m*A \bmod n$ (block 510).

If the TPM does not blind m, then the TPM uses $m' = m$ and $A = 1$ (block 512).

After performing these computations, TPM sends m' to the certifying manufacturer (block 514). The certifying manufacturer computes $c' = m'^{d} \bmod n$ (block 516), and provides c' to the responder (block 518). The TPM of the responder computes $c = c'*B^{-1} \bmod n$ (block 520). Notice that this implies that $c = m^{d} \bmod n$. The values c and m are then stored in the TPM or external storage within the responder (block 522). The pair, c and m, is referred to as a signature of the device manufacturer.

III. FUNCTIONALITY OF THE TPM

The TPM may be adapted to operate in three (3) modes of operation in order for the TPM to prove to a challenger that certain information came from the TPM without revealing information concerning the identity of the TPM or stored cryptographic information. These modes of operation include, but are not limited to: (1) multiple communication mode (Figures 7,8); (2) reduced communication mode (Figure 9); and (3) non-interactive mode (Figure 10).

In general, during multiple communication mode, the TPM sends an output (OUTPUTofROUND) after each iteration (or round) of a direct proof, which requires the challenger to respond with a value (e.g., CHOICE value) for that round. The CHOICE value indicates what information is requested by the challenger for that round. During reduced communication mode, however, the TPM requires the challenger to select and commit to a particular value which will be used as input to the CHOICES for all of the rounds before commencement of the direct proof. Finally, during

non-interactive mode, the TPM computes all of the rounds, and computes CHOICES based on all of the OUTPUTofROUND results for all of the rounds. No interaction with the challenger is required.

As described below, Figures 7-9 illustrate exemplary embodiments of interactive methods for a responder to prove to a challenger that it possesses certain cryptographic information (e.g., a cryptographic parameter such as a signature, secret data, key, etc.) from the certifying manufacturer without revealing the cryptographic parameter. According to one embodiment of the invention and as an illustrative embodiment for clarity purposes, the cryptographic parameter is selected to be a signature.

More specifically, for these embodiments of the invention, a challenger (e.g., first platform 102) supplies an assurance parameter (AP) that indicates the number of iterations (or rounds) of processing to be performed by the TPM. . For Figures 7 and 8, for each round, the challenger will provide a CHOICE value which indicates what type of information is requested by the challenger. This technique requires communications between the TPM and the challenger for each round.

For Figure 9, the challenger commits to a value which will be used to compute the choices for all of the rounds before any of the rounds start. This reduces the number of communications between the TPM and the challenger. Such commitment may be accomplished through a variety of techniques. For instance, according to one embodiment of the invention, the challenger could select a large random or pseudo-random value, RAND, and compute a hash value of RAND, namely HASH(RAND). The challenger could send HASH(RAND) to the TPM. After the TPM has computed

the information from all of the rounds, the TPM could compute a running hash value for the TPM computed information from all of the rounds (referred to as a "RUNNING HASH"). The RUNNING HASH would be sent to the challenger, soliciting the challenger to respond by providing RAND. At this processing stage, the TPM could verify that HASH(RAND) matches a newly computed hash value of RAND. Then, the CHOICEs for all of the rounds could be computed by a combination of RAND and RUNNING HASH, for instance each CHOICE = HASH(RAND, RUNNING HASH) as described below.

Once the CHOICE is revealed by the challenger or determined by the process outlined above, the TPM proves the validity and authenticity of the requested information by providing certain values in accordance with a direct proof as set forth below. This direct proof reduces the overall processing time by the system and substantially reduces the complexity of the communication exchange between the challenger and the responder.

Referring now to Figure 7, prior to performing the direct proof, the TPM receives a value W* from the device manufacturer as part of the cryptographic parameters (block 602). The value W* is chosen to be larger than W. W would typically be chosen to be about 2^{160}, and W* would typically be chosen to be around 2^{180}. However, a different value of W* may be utilized based on the amount of security desired.

The challenger supplies an assurance parameter (AP) to the TPM (block 604). The AP indicates the number of iterations (or rounds) of processing to be performed by the TPM for the subsequent blocks.

As internal operations, the TPM selects y, where $0 < y < n$ (block 606). Such selection may be random or pseudo-random in nature. The TPM computes x such that $x = c*y \bmod n$ (block 608).

The TPM randomly or pseudo-randomly selects S independent values r_i ($1 \leq r\_i \leq W^*$) and computes $r = r\_1 + r\_2 + \ldots + r\_S$. (blocks 610 and 612). Normally, S is a generally small number, less than 100. For one embodiment of the invention, S ranges between 5 and 20. Thereafter, as shown in block 614, the value v is computed ($v = h^r \bmod P$). The value h may be randomly generated, pseudo-randomly generated or generated in a deterministic manner. The value h should have a property that $h^n = 1 \bmod P$ and that the TPM uses a different h value for each challenger. Where the value h is determined randomly or pseudo-randomly, the value h is sent to the challenger, normally prior to the transmission of the cryptographic parameters noted above.

The TPM selects t, where $0 < t < n$ (block 616). Again, such selection may be random or pseudo-random in nature. The value "t" is used to disguise values x^e and y^e after z_x and z_y are computed as shown in equations (1-3) and perhaps known to the challenger. The reason is that the value c can be determined if both x and y are known by the challenger, assuming the challenger also knows the public values, e,n,(blocks 618, 620 and 622).

(1) $z\_x = x^e - t \bmod n$

(2) $z\_y = y^e * r + t \bmod n$

(3) $z = z\_x + z\_y \bmod n$

Thereafter, the TPM individually hashes the values v, y, z_y, z_x, z, t and r to produce hash values Hash(v), Hash(y), Hash(z_y), Hash(z_x), Hash(z), Hash(t) and Hash(r), respectively (block 624). The TPM stores all of these hash values in internal memory 220 of Figure 3 or outputs these hash values into an external memory such as storage unit 206 of Figure 2 (block 626). In addition, the TPM either internally stores y,

y^e, v, r, t or encrypts y, y^e, v, r and t with a storage key (e.g., a private key produced by any selected encryption function) and stores the encrypted values in external memory of the second platform (block 628). It is contemplated that the values of y and t were computed pseudo-randomly from a common random seed so that only y^e, v, r, and the random seed could be stored in order to reduce the amount of storage area required. All other values can be recomputed from these stored values as necessary. The TPM also prompts the responder to send Hash(v), Hash(y), Hash(z_y), Hash(z_x), Hash(z), Hash(t) and Hash(r) to the challenger (block 630). This block of data (Hash(v), Hash(y), Hash(z_y), Hash(z_x), Hash(z), Hash(t), Hash(r)) is the output of the TPM from this round, which is referred to as the "OUTPUTofROUND" as described above. The OUTPUTofROUND is sent to the challenger as each round is computed, and the CHOICE for that round may be provided by the challenger to the TPM at the end of each round. The CHOICE may be revealed through recovery or perhaps generation based on a value originally supplied by the challenger and later value produced by the responder as described above (block 632). For illustrative purposes, the CHOICE is set to have a value ranging between zero (0) and three (3).

If CHOICE = 0, then the TPM sends the values x and t in the clear to the challenger (blocks 634 and 636). This enables the challenger to generally check the value of x because the challenger previously received e & n as part of the cryptographic parameters. Namely, the challenger now computes $z\_x = x^e - t \bmod n$ and verifies the Hash(t) and Hash(z_x).

If CHOICE = 1, then the TPM sends the values y, t, r to the challenger (blocks 638 and 640). This enables the challenger to generally check the value of y. Namely, the challenger computes

$z\_y = y^{e}*r + t \bmod n$ and $v = h^{r} \bmod P$ since cryptographic parameters n, P and h were previously made available to challenger. Thus, challenger verifies the Hash(y) Hash(t), Hash(r), Hash(z_y) and Hash(v). The challenger further verifies that r is within a range from 0 to W*.

If CHOICE = 2, then the TPM sends the values z_x and z_y to the challenger (blocks 642 and 644). This enables the challenger to generally check the value of z. The challenger computes $z = z\_x + (z\_y \bmod n)$ and verifies Hash(z_x) and Hash(z_y).

If CHOICE = 3, then the TPM sends the values y, z, v to the challenger, which enables the challenger to verify Hash(y), Hash(z) and Hash(v) (blocks 646 and 648). The challenger checks that $h^{z} = (k*v)^{\{y\hat{}e \bmod n\}} \bmod P$. The value k should have a property that $k = h^{m} \bmod P$ and is sent to the challenger, normally prior to the transmission of the cryptographic parameters noted above (e.g., with value h).

The challenger computes $s = z*y^{-e} \bmod n$ and checks that s is in a range between Z and Z+W+W*. For CHOICE = 3, it is contemplated that s = m+r and that m+r is in the range from Z to Z+W+W*.

Thus, the only exponentiations required by the TPM is of the form $h^{t} \bmod P$, where t is randomly chosen from a small interval and of the form $y^{e} \bmod n$, where e is a small constant. Previously, the TPM was required to perform exponentiations of the form $h^{t} \bmod P$, where t was the result of a modular exponentiation mod n, and was 2000 bits in length. Thus, for one embodiment of the invention, the computation time to perform the operations of Figure 7 has been reduced by a factor of 5.

Overall, a computation time reduction of at least 50% and even 80% or more is achieved.

Referring now to Figure 8, the same operations associated with the interactive method of Figure 7 are performed. However, the value r is randomly selected (block 670) in lieu of being based on S independent values set forth at blocks 610 and 612 of Figure 7. This requires W* to be of a value of around 2^{220} or higher to get reasonable security.

Referring now to Figure 9, the challenger commits to a value used to compute all of the choices for information (CHOICES) for all of the rounds before any of the rounds commence. In accordance with the TPM being in the reduced communication mode, the number of communications between the TPM and the challenger are greatly reduced.

According to one embodiment of the invention, the TPM computes the value h, which may be randomly generated, pseudo-randomly generated or generated in a deterministic manner. The seed and method for computing the value h may be provided by the challenger. Thereafter, the TPM computes the value k, which is equal to h^m mod P, where "m" is a random number and "P" is a prime number as described above. Both values h and k are sent by the TPM to the challenger (block 702).

The challenger selects a random or pseudo-random value (RAND) which is used to compute CHOICE and provides a modified version of RAND to the TPM. The modified version may be an encrypted version of RAND (e.g., encrypted with a symmetric key shared by the challenger and responder) or a hash result of RAND to produce (HASH(RAND)). The TPM also receives an assurance parameter (AP) from the challenger. The AP indicates the number of iterations (or rounds) of computations to be performed by the

TPM (block 704). A count (i) of the number of rounds performed is set to an initial value (e.g., i=1) and is subsequently adjusted (i.e., incremented or decremented) to ensure that the desired number of rounds are performed.

As internal operations, for each round, the TPM selects y, where $0 < y < n$ (block 706). Such selection may be random or pseudo-random in nature. The TPM computes x such that $x = c*y \bmod n$ (block 708). The TPM randomly or pseudo-randomly selects S independent values r_i ($1 \leq r\_i \leq W^*$) and computes $r = r\_1 + r\_2 + \ldots + r\_S$. (blocks 710 and 712). Normally, S is a generally small number, less than 100. For one embodiment of the invention, S ranges between 5 and 20. Thereafter, as shown in block 714, the value v is computed ($v = h^r \bmod P$). It is noted that the TPM uses a different value h for each challenger.

The TPM randomly or pseudo-randomly selects t, where $0 < t < n$ (block 716). The value "t" is used to disguise values x^e and y^e after z_x and z_y are computed as shown above in equations (1-3). The reason is that the value c can be determined if both x and y are known by the challenger, assuming the challenger also knows the public values, e,n,(blocks 718, 720 and 722).

Thereafter, the TPM individually hashes the values v, y, z_y, z_x, z, t and r to produce hash values Hash(v), Hash(y), Hash(z_y), Hash(z_x), Hash(z), Hash(t) and Hash(r), respectively (block 724). The TPM continuously maintains a running hash value where these hash values are loaded and undergo hash operations (hereinafter referred to as the "RUNNING HASH").

Thereafter, the TPM either internally stores y, y^e, v, r, t or encrypts y, y^e, v, r and t with a storage key (e.g., a private key produced by any selected encryption function) and stores the encrypted values in external memory of the second platform

(block 728). It is contemplated that if values y and t were computed pseudo-randomly from a common random seed, only y^e, v, r, and the random seed would need to be stored in effort to reduce the amount of storage area required.

Thereafter, the count (i) is compared with AP to determine if all rounds have been completed (block 730). If not, the count is adjusted (e.g., incremented) and another round of computations is performed (block 732).

However, if all rounds have been completed, the TPM sends RUNNING HASH to the challenger (block 734). In response the TPM receives RAND from the challenger (block 736). Where the HASH(RAND) was previously sent by the challenger, the TPM verifies RAND by performing a one-way hash operation on RAND using the same hash function as used by the challenger (block 738). This produces a hash value, which is compared with HASH(RAND). If a match is determined, RAND has not been corrupted (block 740). Otherwise, an error is reported (block 742).

Thereafter, the selected CHOICE can be revealed by performing a hash on RAND and RUNNING HASH (block 744). For illustrative purposes, the CHOICE is set to have a value ranging between zero (0) and three (3). The iterative transmission of data is accomplished by a looping function established by blocks 746, 764 and 766. In an alternative embodiment, the CHOICE could be determined directly from RAND.

Herein, as an example, if CHOICE = 0, then the TPM sends the values x and t in the clear to the challenger (blocks 748 and 750). This enables the challenger to generally check the value of x because the challenger previously received e & n as part of the cryptographic parameters. Namely, the challenger

now computes $z\_x = x^e - t \bmod n$ and verifies the Hash(t) and Hash(z_x).

If CHOICE = 1, then the TPM sends the values y, t, r to the challenger (blocks 752 and 754). This enables the challenger to generally check the value of y. Namely, the challenger computes $z\_y = y^e * r + t \bmod n$ and $v = h^r \bmod P$ since cryptographic parameters n, P and h were previously made available to challenger. Thus, challenger verifies the Hash(y) Hash(t), Hash(r), Hash(z_y) and Hash(v). The challenger further verifies that r is within a range from 0 to W*.

If CHOICE = 2, then the TPM sends the values z_x and z_y to the challenger (blocks 756 and 758). This enables the challenger to generally check the value of z. The challenger computes z = z_x + (z_y mod n) and verifies Hash(z_x) and Hash(z_y).

If CHOICE = 3, then the TPM sends the values y, z, v to the challenger, which enables the challenger to verify Hash(y), Hash(z) and Hash(v) (blocks 760 and 762). The challenger checks that $h^z = (k*v)^{\{y\hat{}e \bmod n\}} \bmod P$. The value k should have a property that $k = h^m \bmod P$ and is sent to the challenger, normally prior to the transmission of the cryptographic parameters noted above (e.g., with value h).

The challenger computes $s = z*y^{-e} \bmod n$ and checks that s is in a range between Z and Z+W+W*. For CHOICE = 3, it is contemplated that s = m+r and that m+r is in the range from Z to Z+W+W*.

Of course, it is contemplated that operations set forth in blocks 702 and 704 may be slightly altered to reduce the number of communication cycles between the challenger and the

responder. For instance, when the challenger requests a direct proof from the TPM, the challenger also provides a seed and method for computing the value h, the value of the AP and the modified version of RAND to the responder. The TPM performs the computations for each of the rounds, but at the end, sends h, k and RUNNING HASH to the challenger.

For another embodiment of the invention, referring to Figure 10, an embodiment of a non-interactive method for the responder to prove to a challenger that it has a cryptographic parameter from the certifying manufacturer without revealing the cryptographic parameter is shown. Herein, the direct proof involves a plurality of operations as requested by the challenger, namely based on the value of AP (block 802). After a counter (i) is set, for each iteration, a responder continuously maintains a running hash value based on one or more of the following hash values computed during the iteration: Hash(v), Hash(y), Hash(z_y), Hash(z_x), Hash(z), Hash(t) and Hash(r) (blocks 804, 806, 808, 810, 812). After all (AP) iterations have been completed by the TPM, the running hash value is used to indicate the CHOICE (block 814).

For instance, in one embodiment of the invention, the least significant bits of the running hash value are used to denote which of a plurality of CHOICEs is selected (e.g., two bits needed for 4 CHOICEs). In another embodiment of the invention, the most significant bits of the running hash value are used. In yet another embodiment of the invention, a logical operation is performed on bits of the running hash value to produce a value that denotes one of the plurality of CHOICEs.

While certain exemplary embodiments of the invention have been described and shown in the accompanying drawings, it is to be understood that such embodiments are merely illustrative of

and not restrictive on the broad aspects of various embodiments of the invention, and that these embodiments not be limited to the specific constructions and arrangements shown and described, since various other modifications are possible. It is possible to implement the embodiments of the invention or some of their features in hardware, programmable devices, firmware, software or a combination thereof.